PhUSE 2014

Paper TS08

Manage your SAS Drug Development environment

First author, Bart Van Win, Business & Decision Life Sciences, Brussels, Belgium
Second author, Jean-Marc Ferran, Qualiance, Copenhagen, Denmark
Third author, Sébastien Roland, Business & Decision Life Sciences, Brussels, Belgium

ABSTRACT

When the major players in the pharmaceutical industry start focusing on keeping data management processes internal instead of outsourcing, having a dedicated environment is a plus. SAS brings a new version of the product called SAS Drug Development® (i.e. version 4.x), fulfilling many needs in the search for this platform. This paper will focus on managing a SAS Drug Development 4.x environment.

In the paper following topics will be addressed:

- SDD environmental logic
- Setting up a folder structure with group access and permissions
- Program – Job – Workflow
- Creating privileges and managing users
- Using API functionalities to automate tasks
- Using API functionalities to make user account listings

While discussing these topics, a link will be made with default environments (fileserver, SAS server…) to show the possibilities outside of SDD. A custom application, CDmation, will be introduced as a part of the solution in managing the environment.

INTRODUCTION

This paper will describe the usage and management of an SDD 4.x system. Comparisons with other environments will be made where possible. Sometimes a reference will be made to CDmation, a Business & Decision Life Sciences application.

The core focus will be on administrative tasks, like user and group management, privileges and permissions... There will also be a chapter on the usage of programs, jobs and workflows in relation to the environment. This paper will not focus on how to build programs, jobs and workflows in the system. The last part of the paper will describe how the API functions allow external applications to interact with the environment. To demonstrate the possibilities of these API functions, some examples will be given. One will be a simple example, creating a user listing. The other will be a fully integrated product, CDmation.

What is SDD?
According to SAS, “SAS Drug Development enables the efficient development, execution and management of analysis and reporting activities for clinical research”.

SDD can be seen as fileserver and active directory, versioning system and SAS server, all in one.
SDD can be used to upload, store and version files. The SAS server allows for interaction with them.

The main difficulty is understanding the need of the users. The system will have a variety of users with different roles in the company. Not everyone will require the same functionality from the system. Some will use the system to investigate data, others will just use the system to upload and download files (think about external partners). Some will be programmers and develop SAS programs within SDD, others will use it to store and share documents, with the possibility of versioning an e-signing.

PRIVILEGES AND USER MANAGEMENT

User management in an SDD environment is separate from the companies' environment. The web application keeps its own user database. Having dedicated people manage the system in an administrator role will be necessary. User management consists out of different parts.

1. First there is the user creation, with all the standard user information, like name, e-mail address, account type, etc.
2. The second part is giving the privileges to the users. In table underneath there is an overview of the different privileges. Be aware that these privileges are the global privileges. Later, the paper will describe the privileges within the folder structure (which are different).

Privilege	Description
Access WebDav	Use the SDD DC to access files externally
Create Message	Send a message to another user in the system, the other user can see this message on his/her dashboard
Create SAS Session	Start a SAS Session by opening a SAS program, running a job…
Manage All SAS Sessions	View all SAS Sessions, not only your own. You can also cancel submissions or end SAS sessions
Manage All Schedules	See and change scheduled jobs by other users also
Manage All Subscriptions	See, set and change subscriptions forother users also
Manage Checked out Files	Uncheck checked out files, to for example free up files when a person is on a long leave
Manage Dashboard Welcome Message	Change the Dashboard Welcome Message
Manage Extended Attributes	Change the Extended Attributes for items. Extended Attributes are the extra information on top of the normal file information.
Manage My Schedules	Manage your own schedules
Manage User Account General Information	Manage the general information (e-mail, name…) of a user account.
Manage User Accounts	Manage information like the privileges, lock account…
View Audit History	View the audit history of the system
View Item Audit Records	View the audit history of items, like IDP, folders, files…
Manage Checked out Files	Uncheck checked out files, to for example free up files when a person is on a long leave
Manage Extended Attributes	Change the Extended attributes for items. Extended attributes are the extra information on top of the normal file information
Manage My Schedules	Manage your own schedules

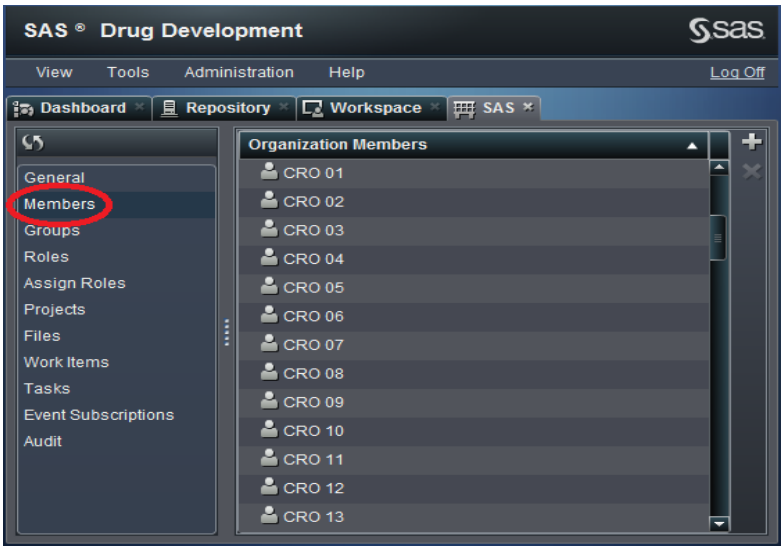
These privileges are set on a user level, not by groups. This can sometimes be a bit cumbersome. For that reason it is a good idea to already sort out the privileges per group, so that if new users require an account, the privileges can be set correctly (example underneath).

	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7	Group 8
Manage Dashboard Welcome Message	x				x			
Manage All SAS Sessions	x				x			
View Audit History	x				x			
Manage My Schedules	x	x	x	x	x			
Manage All Subscriptions	x				x			
Create SAS Session	x	x	x	x	x			
Manage User Account General Information	x							
Manage All Schedules	x				x			
Manage Checked-out Files	x				x			
Access WebDAV	x	x	x	x	x			
Manage Extended Attributes	x				x			
Manage User Accounts	x							
View Item Audit Records	x				x			
Create Message	x	x	x	x	x	x	x	x

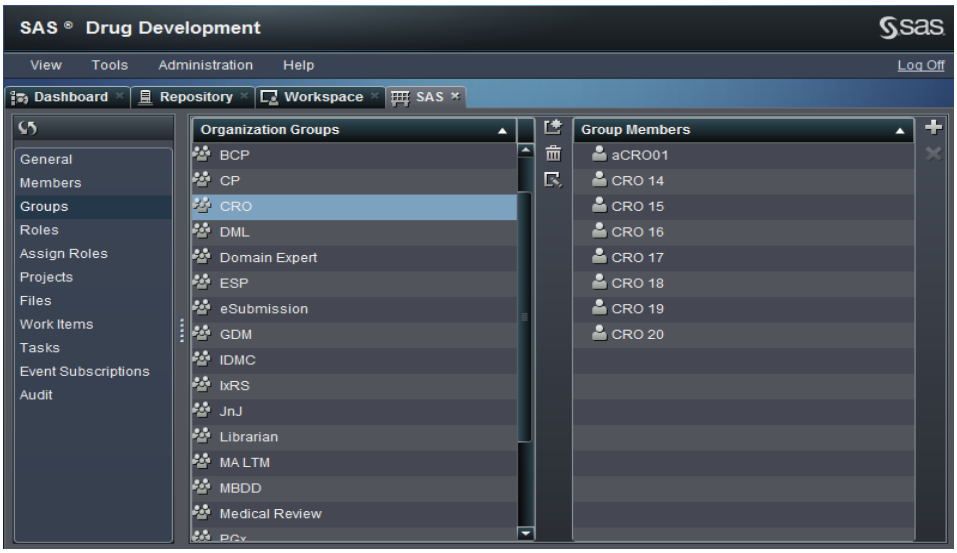
GROUPS

Previous topic mentioned groups in the end. Now the question is; what are groups within SDD? Is it similar to an active directory? The answer is yes… and no. Groups can be used to give folder access (more on this in the next topic), but are handled in a unique way. SDD has multiple levels within the folder structure. Some folders in SDD aren't really seen as folders, but more as manageable objects; objects where you can define groups and link groups to roles and privileges.
These levels are: the root, project and analysis level.
Every level can be seen as its separate environment. Lower levels can inherit groups and roles from higher levels.
Setting up groups in one of the levels can be done as follows:

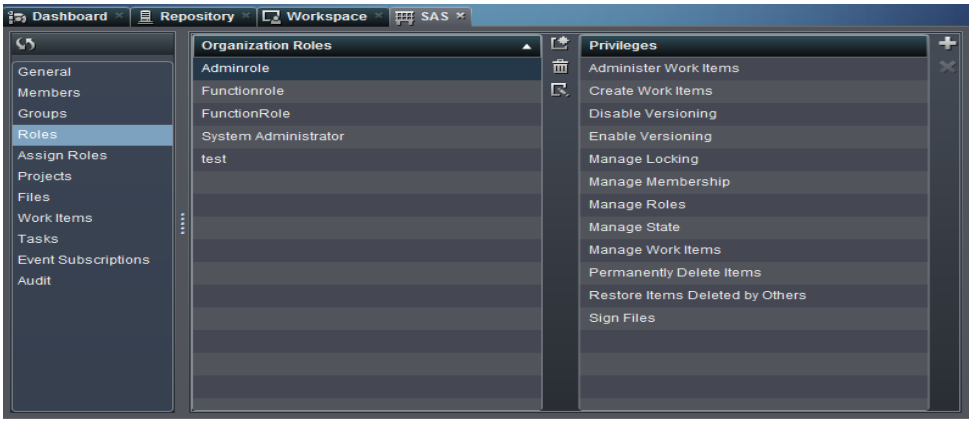
1. Add users as members (only these users will be able to see/use this level, e.g. only the people that have to work on a project are added as members and will be able to see/use the project).



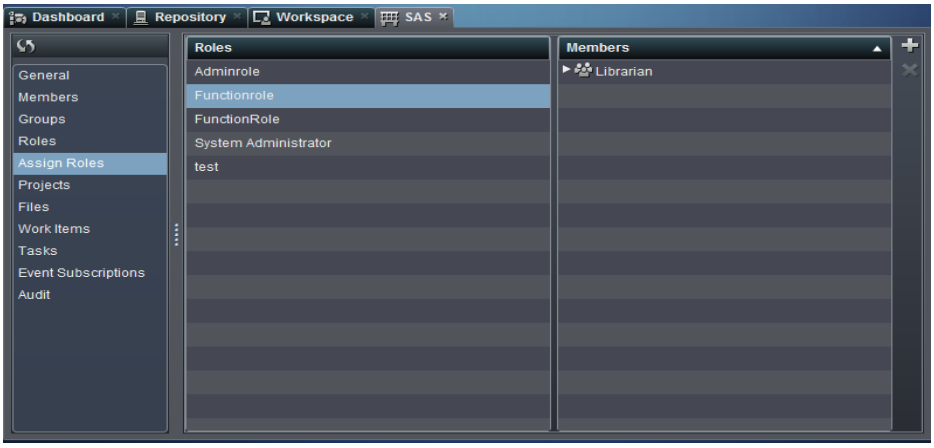
2. Add users to groups.



3. Create roles, which are combinations of privileges. The privileges on these levels are different than the global privileges we have listed before. More information on these privileges will follow.



4. Add groups to roles. Giving groups certain roles, means giving the members of those groups users specific privileges.

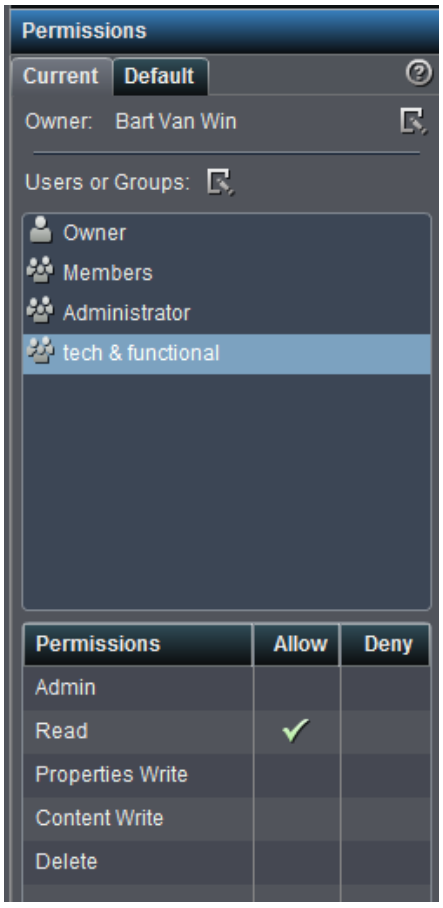


It was mentioned before that these privileges were different. The table underneath gives an overview of these privileges (compare with the previous table to see the differences).

Privilege	Description
Administer Work Items	Admin the workflows in this level
Create Work Items	Create new workflows in this level
Disable Versioning	Allow the user to disable versioning on a file
Enable Versioning	Allow the user to enable versioning on a file
Manage Locking	Manage the locking and unlocking of files
Manage Membership	Manage the groups and members of this level
Manage Roles	Manage the roles of this level
Manage State	Manage the state of this level
Manage Work Items	Manage the workflows in this level
Permanently Delete Items	Remove items deleted by others (from trashcan)
Restore Items Deleted by Others	Restore items deleted by others
Sign Files	Able to sign files

FOLDER ACCESS AND PERMISSIONS

Folder access is given on a group or user basis. Every folder in the repository can be managed by using groups and users from that level (or higher if pass-through). There are five different permissions that can be given to a group or user within a folder. There is the admin, read, properties write, content write and delete permission.
Read and Delete are self-explanatory. Content write means that the user can add and change files. Properties write means that the user can change the properties of a file (this is the metadata of a file).
Every file type has a preset list of properties, but more can be added with external attributes.
The Admin function allows a user to change permissions on a file or folder.



For every folder, there are two different permission sets. There is the current permission set, which allows the user to set the current permission, meaning that if the permissions are changed, the folder and files in the folder (and subfolders if selected) will have the new permissions.
The default permission contains the permissions for the new files in a folder. Files can not have a default permission, because they are not a container.

To clarify this with an example:

1. Folder A, current permission is read/write, default permission is read.
2. A new file is added to Folder A, this file will now only have read permission.
3. The administrator changes the current permission to read/write/delete.
4. Now the file has the read/write/delete permission.

In most cases the current permission should be set equal to the default permission. Sometimes there might be circumstances where they can differ.

PROGRAM – JOB – WORKFLOW

The SDD SAS programming environment is very close to the base SAS one. The difference lies in the implementation. Additional elements are added to the SDD environment: Jobs and workflows.
In SDD a SAS program is the same as a base SAS program. These SAS programs can be added to /used by jobs. Jobs can be set up with parameters that will be used as macro variables within the program. The drawback is that jobs require a lot of metadata to work properly in the repository.
The highest level on top of jobs is workflows. A workflow is an automation framework on top of SDD. Workflows can send out messages, give users tasks, run jobs automatically, etc. Here also, workflows require a lot of metadata setup.

The consensus here is that setting up jobs requires a lot of manual work. This manual work can be simplified though. Jobs can be copied from one study to another, but then will still require additional changes.
The structure of the Job file allows a different approach.
A Job is an XML file. This means that with a “find and replace” (manual) or programmatically, new jobs can be created.

```
- <job releaseVersion="1.0" description="" executionMode="SEQUENTIAL" sddVersion="4.0" log="/SAS
  /Files/ShowFunctionality" lst="/SAS/Files/ShowFunctionality" mnf="/SAS/Files/ShowFunctionality">
  - <tasks>
      <task path="/SAS/Files/ShowFunctionality/program02.sas"/>
    </tasks>
  - <taskSpecs>
      <inputSpec path="/SAS/Files/ShowFunctionality/program02.sas" type="FILE" version="*"
      includeSubFolders="false"/>
    </taskSpecs>
  - <otherSpecs>
      <inputSpec path="/SAS/Files/ShowFunctionality/inputdataset" type="CONTAINER" version="*"
      includeSubFolders="false"/>
    </otherSpecs>
  - <outputPaths>
      <outputPath path="/SAS/Files/ShowFunctionality/outputdataset" includeSubFolders="false"/>
    </outputPaths>
  - <outputSpec enableVersioningForNewFiles="false">
      <versionType type="MAJOR"/>
    </outputSpec>
    <parameters/>
  </job>
```

It is even possible by having a program scan the SAS program(s) and logs, to automatically prepare the job. Scanning of the program can for example be done via running the program with a scaproc procedure at the start and finish.

```
2
3  proc scaproc;
4     record "&path/record.txt";
5  run;
6
```

{{{ run program code }}}

```
2
3  proc scaproc;
4     write;
5  run;
```

API AUTOMATIONS

This topic will consist out of three parts.

1. First, the API functions of SDD will be described.
2. The second part will give a simple example of how you can use JAVA API functions to read out the user management table. In the example an excel table is created with all the necessary user info. The SDD 4.2 system does not have an export functionality build in for this information, but by using the API functions it can be done anyway.
3. The third part will be a more complex example, where Business&Decision Life Sciences has created a fully SDD compatible environment to do Library Management, Study Build and Management, Data Validation and Issue Tracking. The interaction with SDD is relying purely on these JAVA API functions.

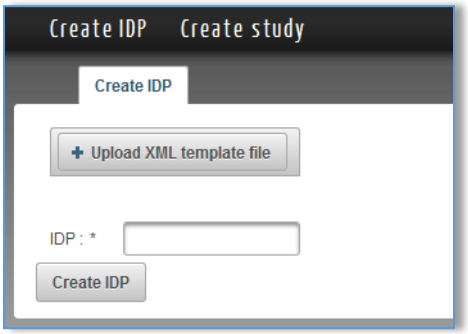
With each version of SDD comes a constantly growing set of API functions, both in JAVA as well as in SAS. The JAVA set contains in each of the SDD versions a bigger list of functions than the SAS set. Documentation is provided for both sets of functions. For JAVA it is stored in the well-known JAVA doc layout. For SAS there is a user's manual in PDF format.
The goal of API functions is to allow the users access to the systems functions, without having to use the system itself. To list the always growing function set, would be to list the whole SDD functionality.
In SDD 4.2 (one of the earlier versions in the 4.x series), the JAVA API functions already allow to create folders and files, read out the audit trail, do user management, run jobs…

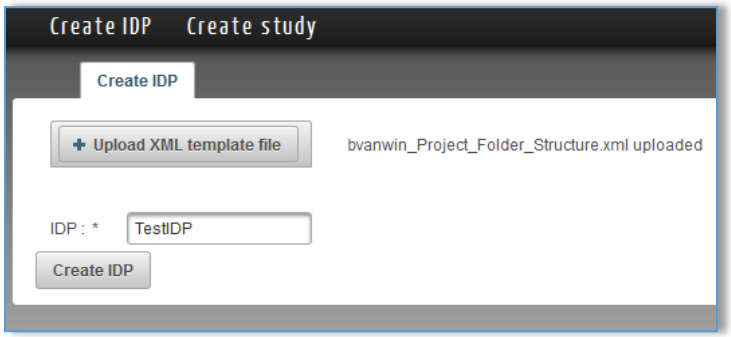
A first easy example of using the JAVA API functions is creating a user listing report in Excel.
This functionality is not possible in the current SDD environment, but can be done via external functions.
The user writes a program that accesses the UserService within SDD and extract the list of users.
With some additional JAVA coding, this set of users can then be transformed into an excel file.
An example of this kind of listing is shown in the image underneath.

	A	B	C	E	F	G	H	J	K
1	FirstNam	LastNam	Departme	IsActive	IsLocked	DisplayNam	LastLogin	AccountType	Administrat
16	User	User	BDLS	Yes	Yes	User User	Thu Jun 13 05:17:41 EDT 2013	INTERACTIVE	No
53	User	User	BDLS	Yes	Yes	User User	Mon Sep 16 09:45:27 EDT 2013	INTERACTIVE	No
58	User	User	BDLS	Yes	Yes	User User	Mon Apr 15 08:18:32 EDT 2013	INTERACTIVE	Yes
68	User	User	BDLS	Yes	No	User User	Wed Sep 24 10:02:23 EDT 2014	INTERACTIVE	No
88	User	User	BDLS	Yes	No	User User	Wed Sep 24 10:57:03 EDT 2014	INTERACTIVE	No
97	User	User	BDLS	Yes	No	User User	Fri Sep 19 10:27:07 EDT 2014	INTERACTIVE	Yes
98	User	User	BDLS	Yes	No	User User	Thu Mar 13 12:02:54 EDT 2014	INTERACTIVE	No
99	User	User	BDLS	Yes	No	User User	Fri Sep 19 08:46:53 EDT 2014	INTERACTIVE	No
103	User	User	BDLS	Yes	Yes	User User	Fri Feb 28 09:49:33 EST 2014	INTERACTIVE	Yes
123	User	User	BDLS	Yes	No	User User	Mon Jul 28 10:20:11 EDT 2014	INTERACTIVE	No

A second and more complicated example can be CDmation, a tool created by Business & Decision Life Sciences as a means to interact with SDD and create a more user friendly environment for the otherwise long manual tasks.
A main functionality is the automatic creation of the folder structure. One of the modules of CDmation reads in XML files and creates new Projects or Analysis with underlying folders accordingly. Groups and roles are also set in the process.

The two images underneath will show the very simple interface, while the images further down give an idea on how the folder structure information can be stored.





Each folder is represented in the XML template

```
<?xml version="1.0" encoding="UTF-8"?>

<folderstructure>
    <folder> <!--/Documentation-->
    <folder> <!--/Documentation/Notifications-->
    <folder> <!--/Documentation/Notifications/DML-->
    <folder> <!--/Documentation/Notifications/GDM-->
    <folder> <!--/Documentation/Notifications/Domain Expert-->
    <folder> <!--/Documentation/Notifications/CRO-->
    <folder> <!--/Documentation/Notifications/PK Office-->
    <folder> <!--/Documentation/Notifications/SD Supplier-->
    <folder> <!--/Documentation/Notifications/Vendor-->
    <folder> <!--/Documentation/Notifications/IxRS-->
    <folder> <!--/Documentation/Notifications/ESP-->
    <folder> <!--/Documentation/Notifications/CP-->
    <folder> <!--/Documentation/Notifications/MBDD-->
    <folder> <!--/Documentation/Notifications/Medical Review-->
    <folder> <!--/Documentation/Notifications/eSubmission-->
    <folder> <!--/Documentation/Notifications/IDMC-->
    <folder> <!--/Documentation/Notifications/BAN-->
    <folder> <!--/Documentation/Notifications/PGx-->
    <folder> <!--/Documentation/Notifications/MA LTM-->
    <folder> <!--/Production-->
```

For each folder, the different groups that will have access + their permissions are set

```
    <folder> <!--/Staging/DM_CRO/Metadata-->
        <name>Metadata</name>
        <path>/Staging/DM_CRO/</path>
        <group>
            <groupname>Administrator</groupname>
            <admin>true</admin>
            <read>true</read>
            <propwrite>true</propwrite>
            <contwrite>true</contwrite>
            <delete>true</delete>
        </group>
        <group>
            <groupname>DML</groupname>
            <admin>false</admin>
            <read>true</read>
            <propwrite>true</propwrite>
            <contwrite>true</contwrite>
            <delete>true</delete>
        </group>
        <group>
        <group>
        <group>
    </folder>
```

The roles, that are managed on the project or analysis level are included

```
<role> <!--Functionrole-->
    <name>Functionrole</name>
    <adminworkitem>true</adminworkitem>
    <createworkitem>true</createworkitem>
    <disablevers>true</disablevers>
    <enablevers>true</enablevers>
    <managelock>false</managelock>
    <managememb>false</managememb>
    <managerole>false</managerole>
    <managestat>false</managestat>
    <managework>true</managework>
    <permdelitems>false</permdelitems>
    <resotreitemsdelbyoth>false</resotreitemsdelbyoth>
    <setsignfiles>true</setsignfiles>
    <assignedto>
        <assignedgroup>Domain Expert</assignedgroup>
        <assignedgroup>GDM</assignedgroup>
        <assignedgroup>DML</assignedgroup>
        <assignedgroup>CP</assignedgroup>
        <assignedgroup>MBDD</assignedgroup>
        <assignedgroup>Medical Review</assignedgroup>
        <assignedgroup>eSubmission</assignedgroup>
        <assignedgroup>MA LTM</assignedgroup>
    </assignedto>
</role>
```

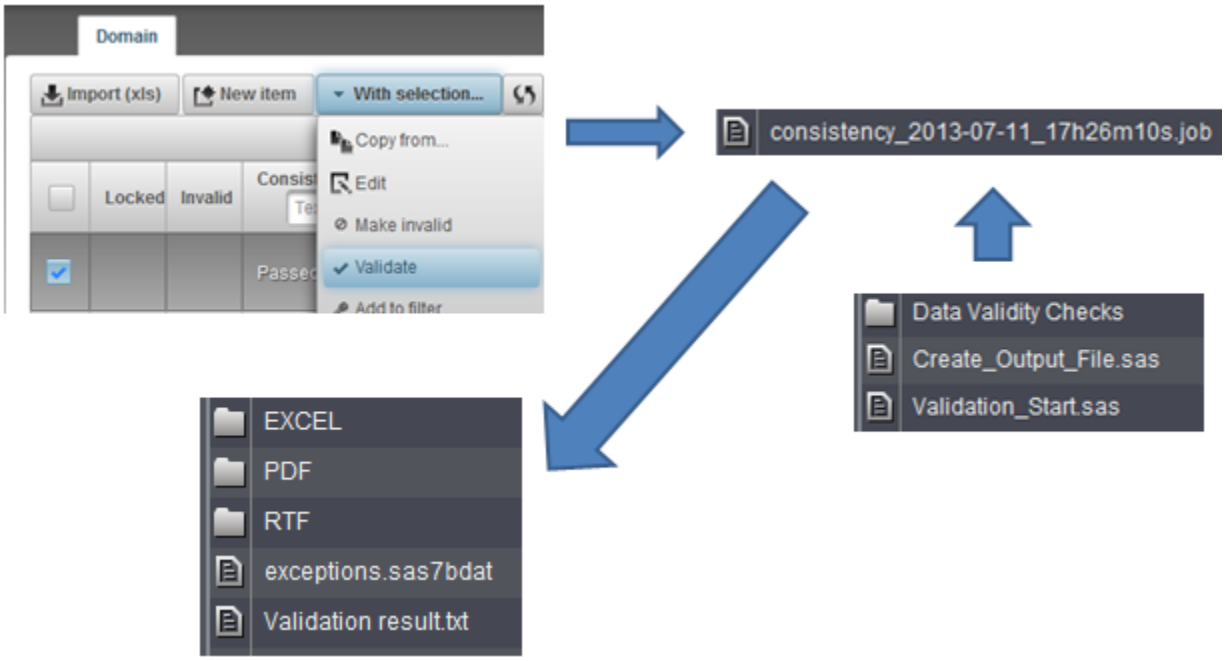
In the end, groups from a higher level can be linked to groups in a lower level, limiting the time needed for administrators when adding users to groups.

```
<group-hierarchy>
    <group-pair>
        <source>DML</source>
        <target>DML</target>
    </group-pair>
    <group-pair>
        <source>Administrator</source>
        <target>Administrator</target>
    </group-pair>
    <group-pair>
        <source>Domain Expert</source>
        <target>Domain Expert</target>
    </group-pair>
    <group-pair>
        <source>GDM</source>
        <target>GDM</target>
    </group-pair>
</group-hierarchy>
```

Another important functionality of the tool is the automatic creation of jobs. Creating jobs in the system can be very time consuming and this is time better spent elsewhere. By using the API functions, the JAVA application creates the job with the correct parametes, inputs and outputs. In the example given underneath (pictures), the job is validating elements in an oracle library.

The user has an element selected in the user interface and selects validate. The application sets the validation status in the oracle database on pending. After this, the application will build the job and start the job. At the end of the job the validation result is written to the database, at which point the JAVA application picks up the result and demonstrates it in the user interface.

Image showing the functionality of the Validation Job creator.



CONCLUSION

The SAS Drug Development 4.x environment is a strong addition to the SDD series. The SAS component lies much closer to the SAS Base environment that everyone is familiar with, in comparison with older versions. Additional to the SAS Base programs, the system provides jobs and workflows as overlaying elements. The file server component gives a lot of possibilities together with the user and group level privileges and permissions.
Where functionalities within SDD are missing, the user is free to develop external applications. Via these applications the user will be able to interact with the system with more functionalities than the standard SDD interface brings.

CONTACT INFORMATION

Contact the author at:
Bart Van Win
Business & Decision Life Sciences
St Lambertusstraat 141 Rue Saint-Lambert
1200 Brussel – Bruxelles
Work Phone: +32 479 707 626
Email: bart.vanwin@businessdecision.com
Web: www.businessdecision-lifesciences.com

Contact the author at:
Sébastien Roland
Business & Decision Life Sciences
St Lambertusstraat 141 Rue Saint-Lambert
1200 Brussel – Bruxelles
Work Phone: +32 xxx xxx xxx
Email: sebastien.roland@businessdecision.com
Web: www.businessdecision-lifesciences.com

Contact the author at:
Jean-Marc Ferran
Business & Decision Life Sciences
St Lambertusstraat 141 Rue Saint-Lambert
1200 Brussel – Bruxelles
Work Phone: +xx xxx xxx xxx
Email: jean-marc.ferran@businessdecision.com
Web: www.businessdecision-lifesciences.com